Lab - 1

Study of Cloud Computing & Architecture

DeepSphere.AI
Enterprise AI and IIoT for Analytics

Google Cloud Platform Services

AIM

- Setting up an account with a cloud provider (AWS, Azure, or GCP)
- Creating virtual machines
- managing storage

Steps For Creating the GCP Account :

1. Go to the Google Cloud Platform website (cloud.google.com)
2. Click the "Try It Free" button to start the sign-up process.
3 .Enter your email address and create a password to create a Google account, or sign in with an existing account.
4. Verify your email address by clicking on the verification link sent to your email.
5. Complete the registration form with your personal information and billing details.
6. Accept the terms of service and privacy policy.

7. Click the "Start My Free Trial" button to create your account and start using Google Cloud Platform services.
8. Add a credit card or link a bank account for billing purposes.
9. Set up a billing alert to notify you when you're approaching your budget limit.

Creating a Virtual Machine in GCP:

1. Log in to the Google Cloud Console (console.cloud.google.com)
2. Click on the "Compute Engine" option in the navigation menu.
3. Click on the "Create" button to create a new virtual machine.
4. Choose a name and region for your virtual machine.
5. Select the machine type and number of vCPUs and memory.
6. Choose an operating system and boot disk.
7. (Optional) Configure firewall and network settings.
8. Click on the "Create" button to create the virtual machine.
9. Once the virtual machine is created, you can access it via SSH or RDP by clicking on the "Connect" button in the virtual machine details page.

Note: It is also possible to create a virtual machine using the Gcloud command-line tool or through the API.
Also, it is recommended to have a project created in GCP before creating a Virtual Machine.

Managing Storage in GCP:

1. Log in to the Google Cloud Console (console.cloud.google.com)
2. Click on the "Storage" option in the navigation menu.
3. Click on the "Create Bucket" button to create a new storage bucket.
4. Choose a unique name for your bucket, and select a storage class and location.
5. (Optional) Configure access controls and encryption settings.
6. Click on the "Create" button to create the storage bucket.
7. Once the bucket is created, you can upload, download, and manage files and objects in the bucket through the Cloud Console, gsutil command-line tool, or the Cloud Storage API.
8. You can also create and manage persistent disks, which are block storage device that can be attached to virtual machines.
9. To create a persistent disk, go to the "Compute Engine" section of the Cloud Console, and click on the "Create" button.
10. Give it a name, select the zone, size, and type of the disk.
11. (Optional) Encrypt the disk, set up automatic snapshot schedule.
12. Click on the "Create" button to create the persistent disk.
13. It is also important to monitor the storage usage and costs on a regular basis, and to set up alerting and budgeting to avoid unexpected charges.

Lab - 02

Virtualization in Cloud by using KVM and VMware

Virtualization in Cloud by using KVM and VMware

AIM

Virtualization is the process of creating a virtual version of a computing resource, such as a operating system, a server, a storage device or network resources. KVM (Kernel-based Virtual Machine) is an open-source virtualization technology for Linux.

Steps :

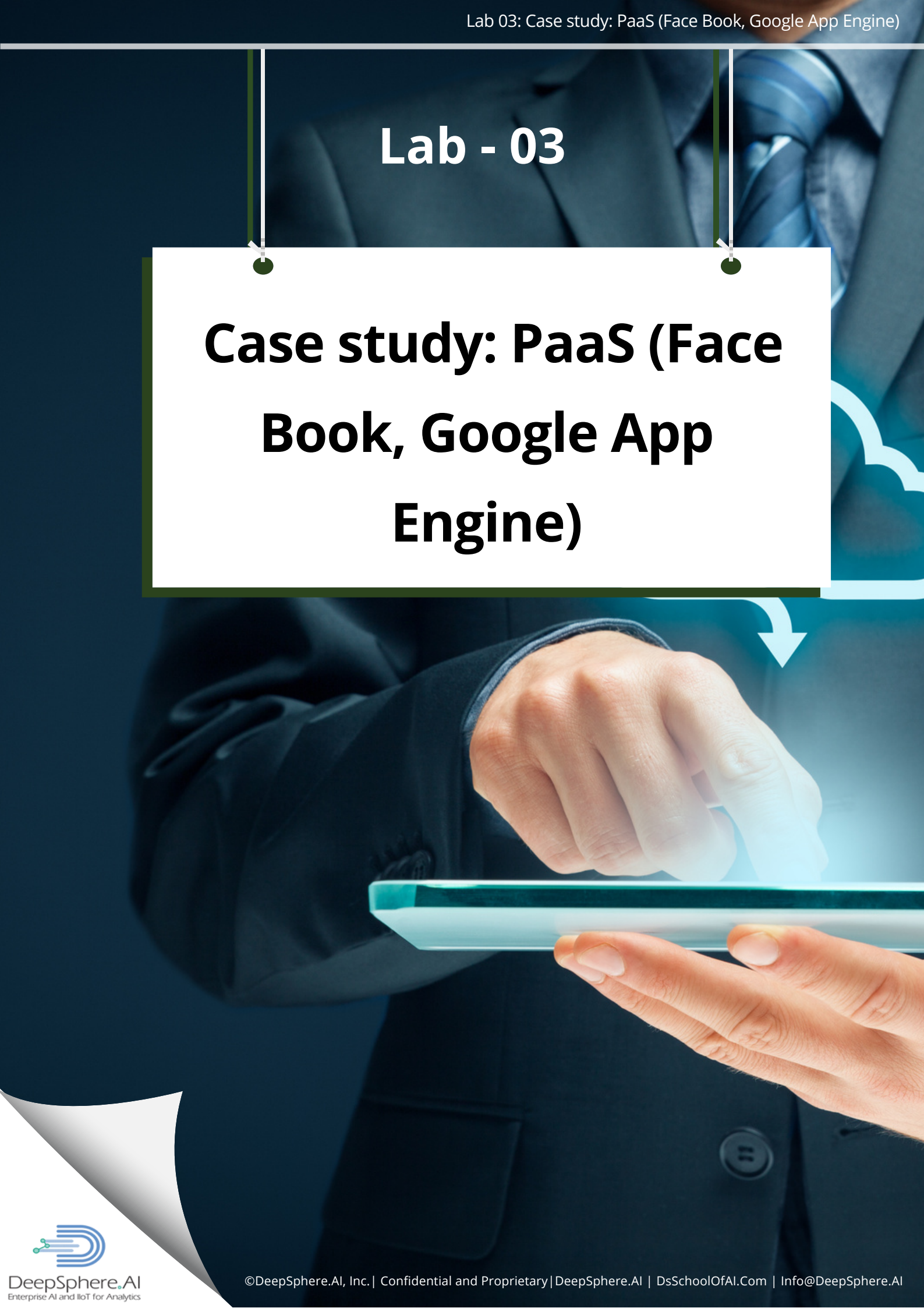
1. Select a host machine that will run the virtualization software.
2. Install the virtualization software of your choice on the host machine.
3. Install an operating system on the virtual machine.
4. Configure the network settings for the virtual machine.
5. You can start it up and begin (as physical machine)
6. To manage the virtual machine, you can use the virtualization software's management tools.

KVM and VMware:

VMware and KVM (Kernel-based Virtual Machine) are both virtualization technologies. VMware is a proprietary software that creates and runs virtual machines on x86-64 computers. KVM, on the other hand, is an open-source virtualization technology that uses the Linux kernel's virtualization capabilities to create and run virtual machines. Both VMware and KVM can be used to create and manage virtual machines, but they have different features and capabilities.

Lab - 03

Case study: PaaS (Face Book, Google App Engine)



Case Study: PaaS at Facebook and Google App Engine

Introduction:

Platform as a Service (PaaS) is a cloud computing model that allows developers to build, test, and deploy applications without the need for managing the underlying infrastructure. Facebook and Google are two companies that have successfully implemented PaaS solutions for their own use and for external developers.

Facebook:

Facebook has developed its own PaaS solution called the Facebook Platform, which allows developers to build and deploy applications on the Facebook website. The platform provides a set of APIs and tools that allow developers to access the social network's user data, create custom experiences, and integrate with other services. One of the key benefits of the Facebook Platform is the large user base and the ability to target specific demographics.

Google App Engine:

Google App Engine is a PaaS offering from Google that allows developers to build and deploy web applications on the Google Cloud Platform. The service provides a set of tools and APIs that allow developers to easily scale and manage their applications. App Engine also provides automatic scaling and load balancing, making it easy to handle large amounts of traffic. Additionally, App Engine offers various features such as NoSQL datastore, Memcache, Task Queues, Cron Jobs and supports various programming languages like Python, Java, Go, PHP.

Comparison:

Both Facebook and Google App Engine provide a PaaS solution for developers, but they have some key differences. Facebook's platform is tightly integrated with the social network and provides access to user data and the ability to target specific demographics. On the other hand, Google App Engine is a more general-purpose PaaS solution that can be used for a wide range of applications and supports multiple programming languages. Facebook Platform is more suited for social media and networking applications, while Google App Engine is more suited for web applications that need to handle large amounts of traffic.

Conclusion:

Facebook and Google App Engine are both successful examples of PaaS solutions that provide developers with the tools and infrastructure needed to build and deploy applications. The Facebook platform is tightly integrated with the social network, providing access to user data and the ability to target specific demographics. On the other hand, Google App Engine is a more general-purpose PaaS solution that can be used for a wide range of applications and supports multiple programming languages. Both platforms have their own advantages and can be used for different types of applications.

Lab - 04

Case Study on Amazon Web Services

DeepSphere.AI
Enterprise AI and IIoT for Analytics

Case Study: AWS at Netflix

Introduction:

Amazon Web Services (AWS) is a collection of remote computing services (also called web services) that make up a cloud computing platform, offered by Amazon.com. Netflix is a company that has successfully implemented AWS to support its streaming video service.

Background:

Netflix is a streaming service that allows users to watch movies and television shows on a variety of devices. The company has been using AWS since 2008 to support its streaming service. Netflix began by using AWS for simple tasks such as data storage and content delivery, but over time it has grown to rely on the platform for almost all of its infrastructure needs.

AWS Services used:

Netflix makes use of a wide range of AWS services to support its streaming service, including:

- Elastic Compute Cloud (EC2) for virtualized servers
- Simple Storage Service (S3) for storage and content delivery
- Elastic Block Store (EBS) for storage of critical data
- Elastic Load Balancing (ELB) for distributing traffic across multiple servers
- CloudWatch for monitoring and logging
- Route 53 for DNS and domain management

Benefits:

AWS has provided Netflix with a number of benefits including:

Scalability: AWS has allowed Netflix to easily scale its infrastructure as needed to handle fluctuations in traffic.

Availability: Netflix has been able to use AWS to ensure high availability for its streaming service.

Cost Savings: Netflix has been able to save money by using AWS instead of building and maintaining its own data centers.

Conclusion:

Netflix is a prime example of how a company can successfully implement AWS to support its operations. The streaming service relies on a wide range of AWS services, including EC2, S3, EBS, ELB, CloudWatch, and Route 53, to support its streaming service. By using AWS, Netflix has been able to achieve scalability, availability, and cost savings. AWS has allowed Netflix to grow its streaming service to millions of users around the world, making it one of the most popular streaming services available today.

Lab - 05

Case Study on Microsoft Azure

Case Study: Azure at Heinz

Introduction:

Microsoft Azure is a cloud computing platform and service that provides a variety of services for building, deploying, and managing applications and services. Heinz, a food and beverage company, has successfully implemented Azure to support its operations.

Background:

Heinz is a global food and beverage company that produces a wide range of products, including ketchup, sauces, and baby food. The company has been using Azure since 2016 to support its operations, specifically to power its supply chain and logistics operations.

Azure Services used:

Heinz makes use of a number of Azure services to support its operations, including: Azure IoT Hub for connecting and managing IoT devices. Azure Machine Learning for predictive analytics. Azure Stream Analytics for real-time data processing. Azure Data Factory for data integration and ETL. Azure Event Grid for event-driven architecture. Azure Active Directory for identity and access management

Azure has provided Heinz with a number of benefits including:

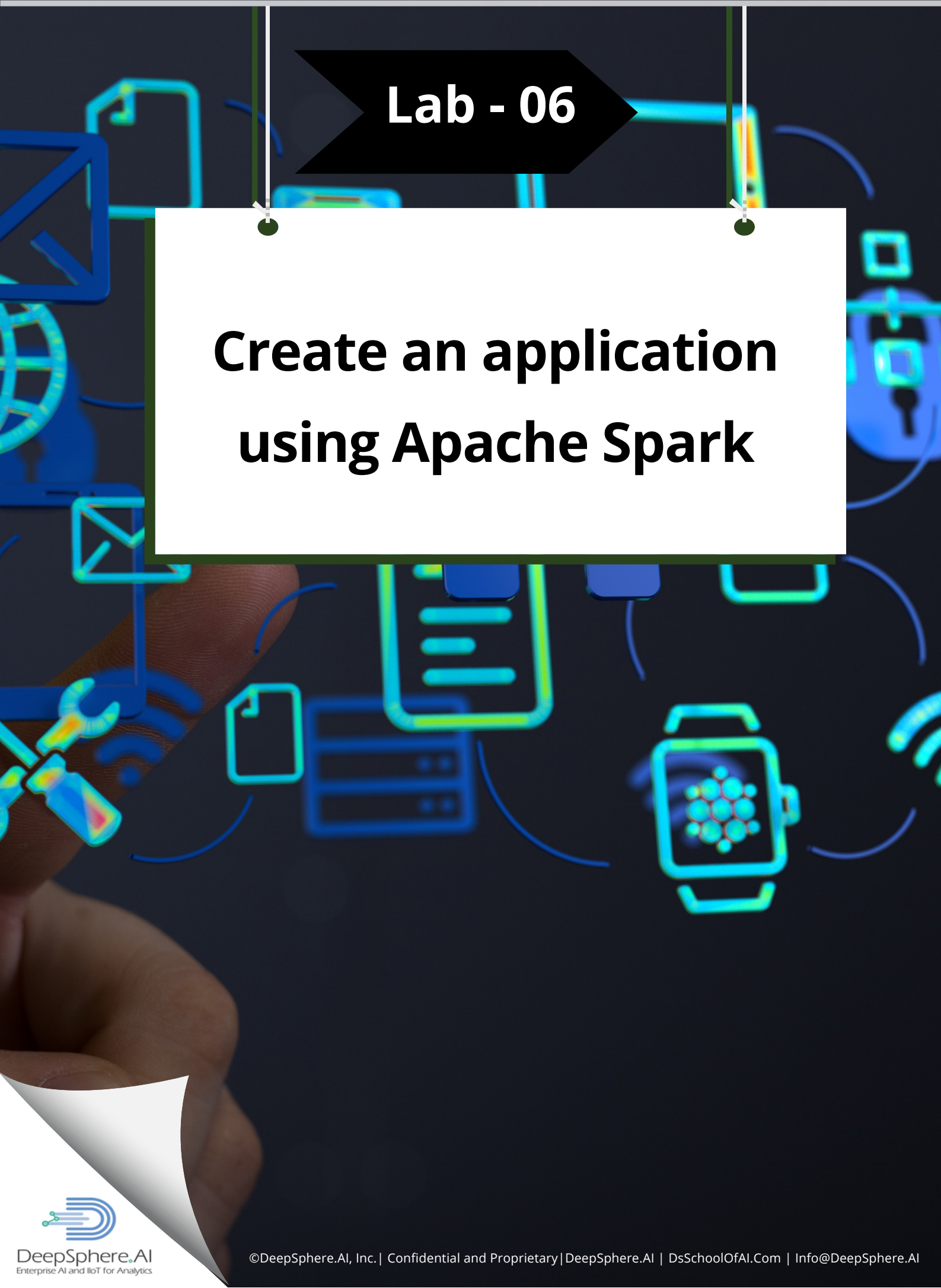
Real-time visibility: By using Azure IoT Hub, Heinz has been able to gain real-time visibility into its supply chain and logistics operations.

Predictive analytics: Azure Machine Learning has allowed Heinz to perform predictive analytics on its data, providing insights that have helped the company optimize its operations.

Efficiency: Azure has allowed Heinz to automate various processes, making its operations more efficient.

Conclusion:

Heinz is a prime example of how a company can successfully implement Azure to support its operations. The food and beverage company relies on a number of Azure services, including IoT Hub, Machine Learning, Stream Analytics, Data Factory, Event Grid, and Active Directory, to power its supply chain and logistics operations. By using Azure, Heinz has been able to achieve real-time visibility, predictive analytics, and efficiency. Azure has allowed Heinz to optimize its operations and improve its bottom line.



Lab - 06

Create an application using Apache Spark

AIM

To Create an application using Apache Spark

Steps :

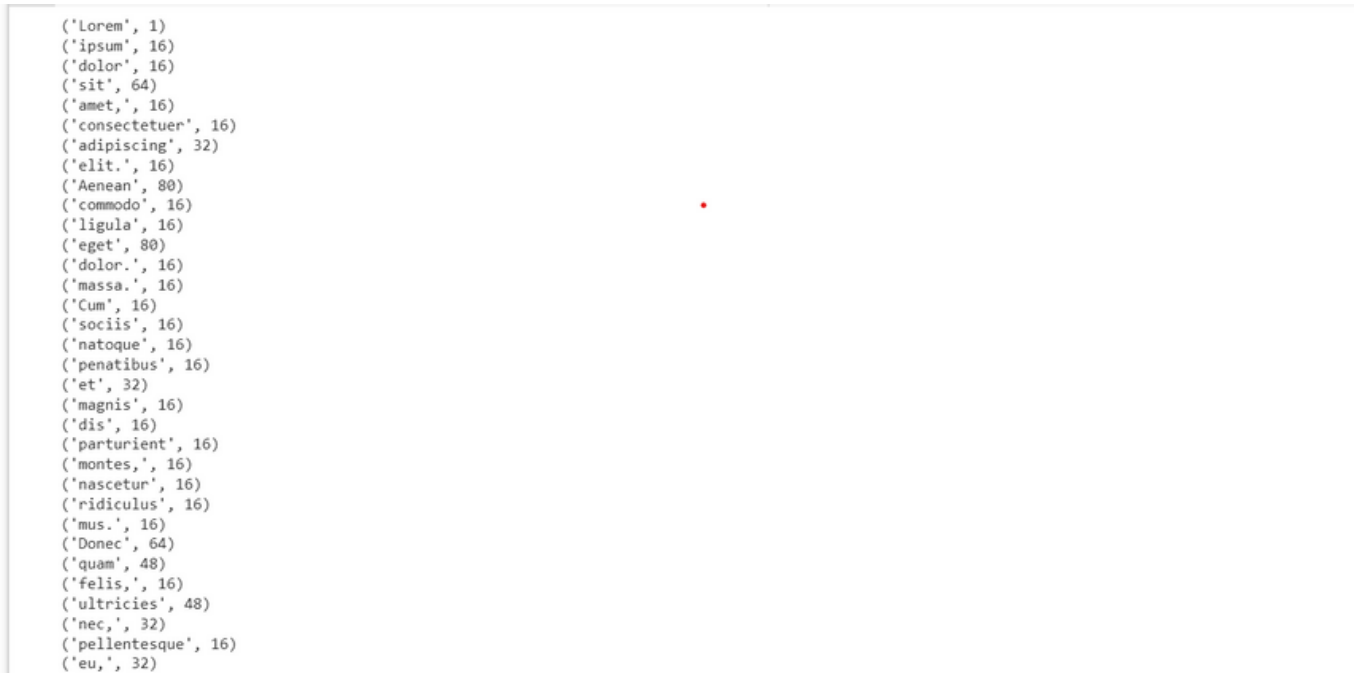
1. Import the necessary modules from PySpark, including the SparkConf and SparkContext classes.
2. Set up the Spark configuration and context. This includes setting the app name and creating an instance of the SparkContext class.
3. Read in the two documents as Resilient Distributed Datasets (RDDs) using the textFile() method of the SparkContext class.
4. Perform word count on the documents by splitting the text into words, mapping each word to a tuple of (word, 1), and then reducing the tuples by key to get the count of each word.
5. Perform a join on the word count RDDs to find the common words and their counts.
6. Print the result using the collect() method of the RDD.
7. Optionally, you can save the results to a file or a database

Program : Finding word count

```
from pyspark import SparkConf, SparkContext
import pandas as pd

# Set up the Spark configuration and context once
conf = SparkConf().setAppName("SimilarityWordCount")
sc = SparkContext(conf=conf)
```

```
# Read in the two documents as RDDs
doc1 = sc.textFile("sample.txt")

# Perform word count on the documents
wordCount1 = doc1.flatMap(lambda line: line.split(" ")) \
                 .map(lambda word: (word, 1)) \
                 .reduceByKey(lambda a, b: a + b)

# Perform a join on the word count RDDs to find the common words and their counts
commonWordCount = wordCount1

# Print the result
a=commonWordCount.collect()
for i in a:
  print(i)
```

Output :



```
('Lorem', 1)
('ipsum', 16)
('dolor', 16)
('sit', 64)
('amet,', 16)
('consectetuer', 16)
('adipiscing', 32)
('elit.', 16)
('Aenean', 80)
('commodo', 16)
('ligula', 16)
('eget', 80)
('dolor.', 16)
('massa.', 16)
('Cum', 16)
('sociis', 16)
('natoque', 16)
('penatibus', 16)
('et', 32)
('magnis', 16)
('dis', 16)
('parturient', 16)
('montes,', 16)
('nascetur', 16)
('ridiculus', 16)
('mus.', 16)
('Donec', 64)
('quam', 48)
('felis,', 16)
('ultricies', 48)
('nec,', 32)
('pellentesque', 16)
('eu,', 32)
```

Lab - 07

Perform a Simple Vector Addition using OpenMP Programming

AIM

Perform a Simple Vector Addition using OpenMP Programming using C.

Program

```c
int main() {
    int i;
    double a[N], b[N], c[N];
    // Initialize the vectors
    for (i = 0; i < N; i++) {
        a[i] = i;
        b[i] = 2 * i;
    }
    // Perform the vector addition using OpenMP
    #pragma omp parallel for
    for (i = 0; i < N; i++) {
        c[i] = a[i] + b[i];
    }
   // Print the result
    for (i = 0; i < N; i++) {
        printf("c[%d] = %f\n", i, c[i]);
    }
    return 0;
}
```

Output :

```
c[0] = 0.000000
c[1] = 3.000000
c[2] = 6.000000
c[3] = 9.000000
c[4] = 12.000000c[5] = 15.000000
c[6] = 18.000000
c[7] = 21.000000
c[8] = 24.000000
c[9] = 27.000000
c[10] = 30.000000
c[11] = 33.000000
c[12] = 36.000000
c[13] = 39.000000
c[14] = 42.000000
c[15] = 45.000000
c[16] = 48.000000
c[17] = 51.000000
```

Lab-08

Write a MPI Program to send data across all processes

AIM

To write a MPI Program to send data across all processes.

Program

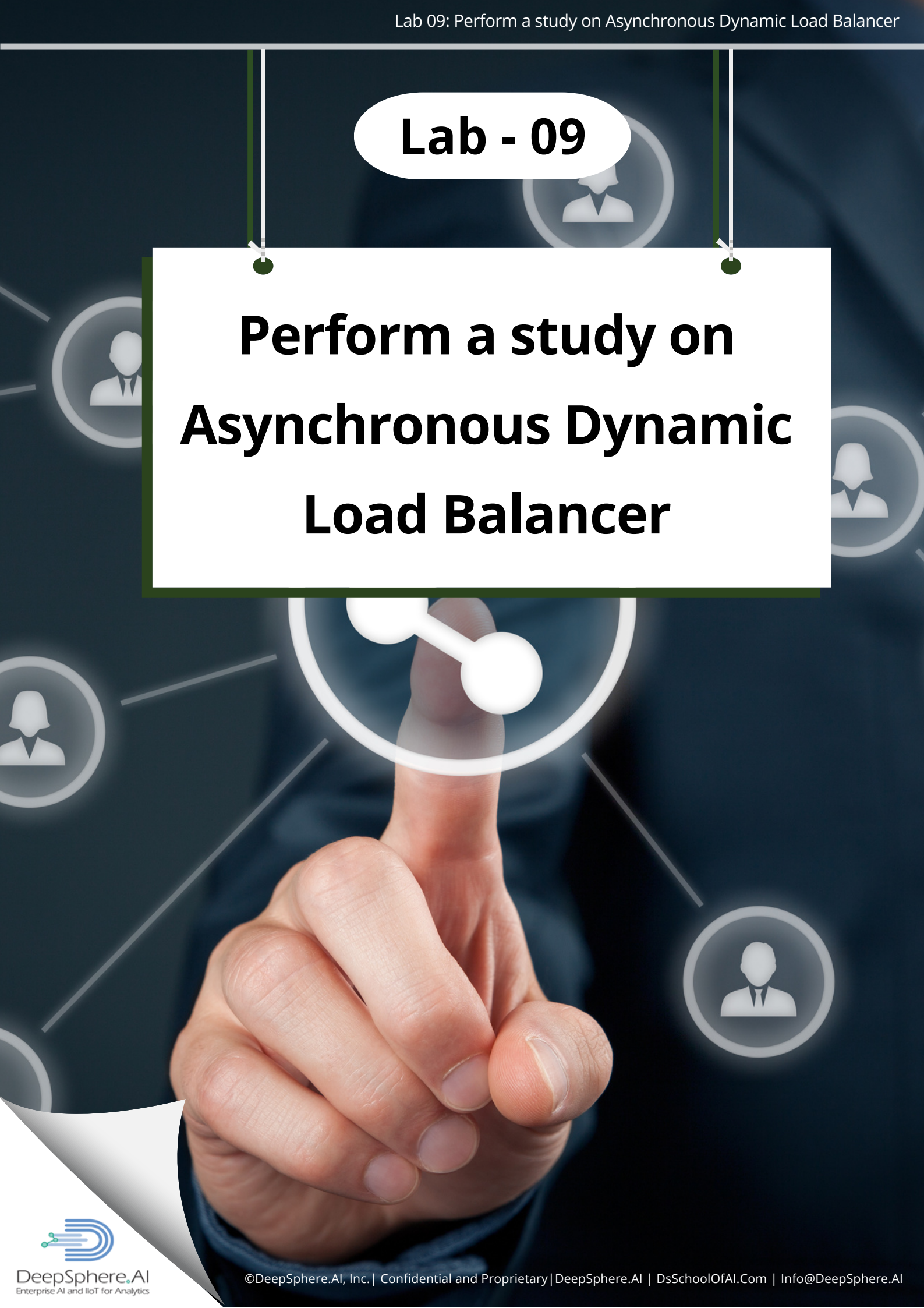
```
#include <stdio.h>
int main(int argc, char *argv[]) {
    int rank, size, data;

    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    if (rank == 0) {
        data = 42;
        printf("Process %d sending data %d to all other processes\n", rank, data);
    }

    MPI_Bcast(&data, 1, MPI_INT, 0, MPI_COMM_WORLD);

    printf("Process %d received data %d\n", rank, data);

    MPI_Finalize();
    return 0;
}
```

Output

```
Process 0 received data 100
Process 1 received data 100
Process 2 received data 100
Process 3 received data 100
```

Lab - 09

Perform a study on Asynchronous Dynamic Load Balancer



DeepSphere.AI
Enterprise AI and IIoT for Analytics

Perform a study on Asynchronous Dynamic Load Balancer

Aim

To perform a study on asynchronous dynamic load balancer.

Introduction

Asynchronous dynamic load balancer is a method used to distribute workloads among multiple processing units in a distributed computing system. It is based on the principle of asynchrony, which allows the processing units to operate independently of each other, and the dynamic nature of the load balancing algorithm, which adapts to changing workloads and system conditions.

Advantage

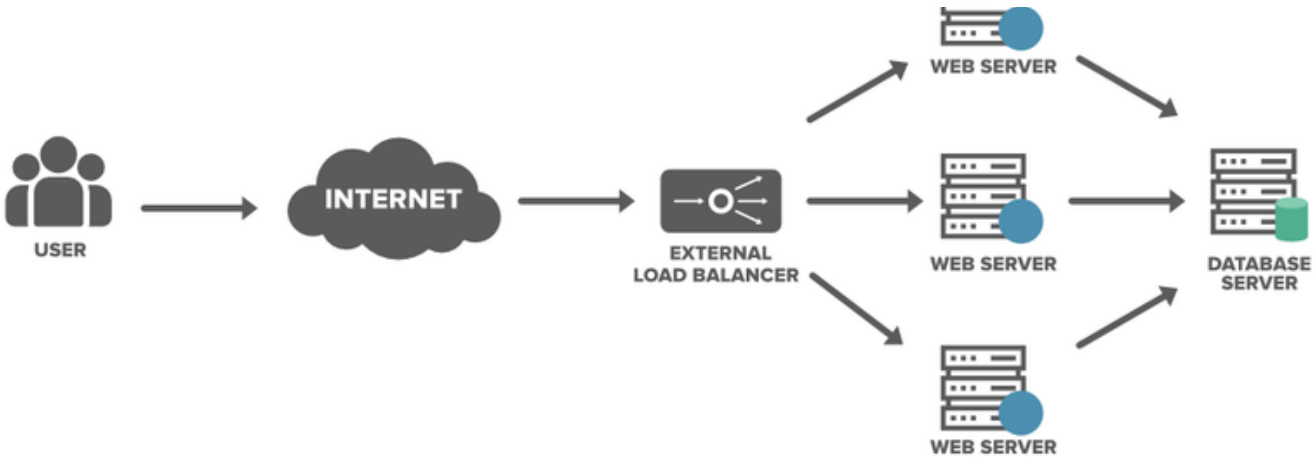
The main advantage of using an asynchronous dynamic load balancer is improved scalability and fault tolerance. Because the processing units operate independently, they can continue processing tasks even if one or more units fail. Additionally, the dynamic nature of the algorithm allows the system to adapt to changes in workload and system conditions, which helps to ensure that the resources are being used efficiently.

Types of algorithms

There are different types of algorithms that can be used for asynchronous dynamic load balancing. One popular approach is the use of work stealing, where idle processing units steal work from busy units. Another approach is the use of message passing, where tasks are sent to idle units through inter-process communication. A study of asynchronous dynamic load balancing can involve comparing the performance of different algorithms under various workloads and system conditions.

This can include measuring the time taken to complete a task, the number of tasks completed, and the overall efficiency of the system. The study can also look at the scalability and fault tolerance of the system, and how well it adapts to changes in workload and system conditions.

In addition, the study can evaluate how the Asynchronous dynamic load balancing algorithms perform in comparison with other load balancing algorithms such as static load balancer, dynamic load balancer, and Hybrid load balancer. Overall, an asynchronous dynamic load balancer can be a powerful tool for improving the scalability and fault tolerance of a distributed computing system. Further studies are needed to evaluate the performance of different algorithms under various workloads and system conditions, and to identify the best practices for implementing such load balancer in practice.



Lab - 10

Perform a study on Parallel Meshing

Introduction

Parallel meshing is a technique used in numerical simulations, particularly in the field of computational fluid dynamics (CFD), to create a mesh that can be distributed and solved on multiple processors or computing nodes in parallel. This allows for faster and more efficient simulations on high-performance computing systems. The mesh is typically divided into smaller sub-domains, which can be solved independently and then combined to form the final solution. This approach is also known as domain decomposition.

Case study on Parallel Meshing

- A parallel meshing case study typically involves analyzing the performance of a mesh generation algorithm when it is executed on multiple processors or cores at the same time. The main goals of the study are to identify bottlenecks in the algorithm that limit its parallel performance, and to develop strategies to overcome these bottlenecks.
- One example of a parallel meshing case study is the parallelization of a 3D Delaunay mesh generation algorithm. The study found that the algorithm had poor parallel performance due to its use of a global priority queue, which required frequent communication between processors. To improve performance, the researchers implemented a parallel version of the priority queue that reduced the amount of communication required.
- Another example is a parallel version of the 2D advancing front method for generating triangular meshes. The study found that the algorithm had poor performance due to the large number of edge flips required, which were performed sequentially. The researchers implemented a parallel version of the algorithm that performed the edge flips in parallel, resulting in significant speedup.
- In general, parallel meshing case studies can be applied to a wide range of mesh generation algorithms and can lead to significant improvements in performance and scalability.

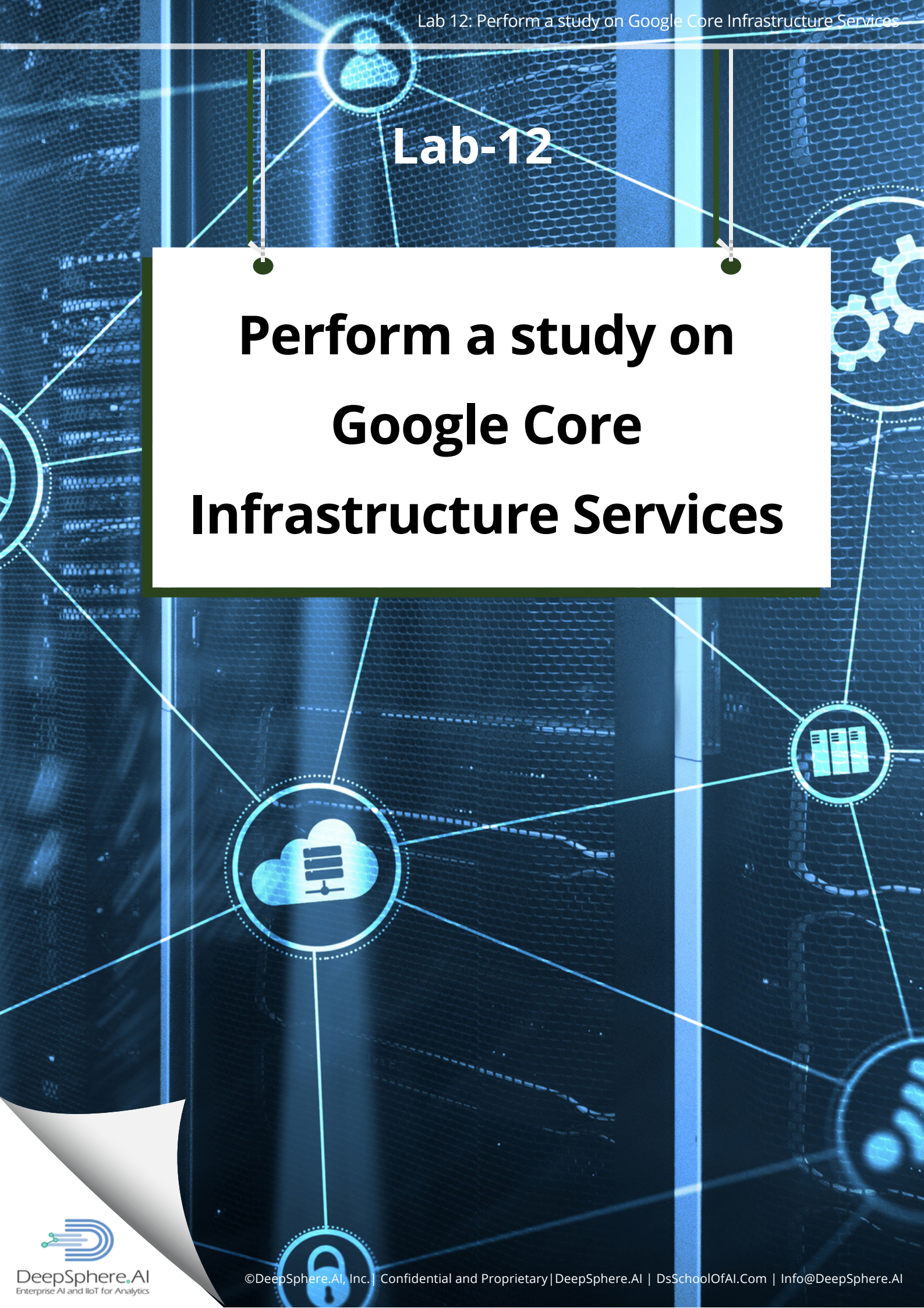
Lab - 11

Perform a study on Networking and Storage Service

DeepSphere.AI
Enterprise AI and IIoT for Analytics

To Perform a study on Networking and Storage Service

- A case study on networking and storage services would typically involve analyzing the performance and scalability of a specific network and storage infrastructure, such as a cloud-based service or a private data center. The main goals of the study would be to identify bottlenecks and limitations in the current infrastructure and to develop strategies to improve performance and scalability.
- One example of a case study on networking and storage services is the implementation of a distributed storage system for a large-scale video streaming service. The study found that the existing centralized storage system was not able to handle the large number of concurrent users and the high volume of data traffic. To improve performance, the researchers implemented a distributed storage system that used multiple servers and a content delivery network (CDN) to distribute the data traffic across multiple locations.



Lab-12

Perform a study on Google Core Infrastructure Services

Perform a study on Google Core Infrastructure Services

Google Core Infrastructure Services are a set of foundational services that provide the foundation for many of Google's other services. These services include:

1. **Compute Engine:** This service allows users to create and run virtual machines on Google's infrastructure. Users can choose from a variety of machine types and configure their own custom machine types. Compute Engine supports Linux and Windows operating systems and offers a variety of storage options including standard storage, SSD storage, and local SSD storage.
2. **Kubernetes Engine:** This service allows users to deploy, scale, and manage containerized applications using Kubernetes, an open-source container orchestration system. Kubernetes Engine provides automatic scaling, self-healing, and load balancing for applications, making it easy to manage and operate containerized applications.
3. **Cloud Storage:** This service provides a scalable, durable, and secure storage solution for data and files. Cloud Storage supports a variety of file types, including binary data, text, and images. It offers both object storage and block storage options, and provides automatic data replication across multiple locations for increased durability and availability.
4. **Cloud SQL:** This service allows users to create and manage relational databases in the cloud. Cloud SQL supports a variety of database engines including MySQL, PostgreSQL, and SQL Server. It provides automatic backups, automatic replication, and automatic failover for increased reliability and availability.

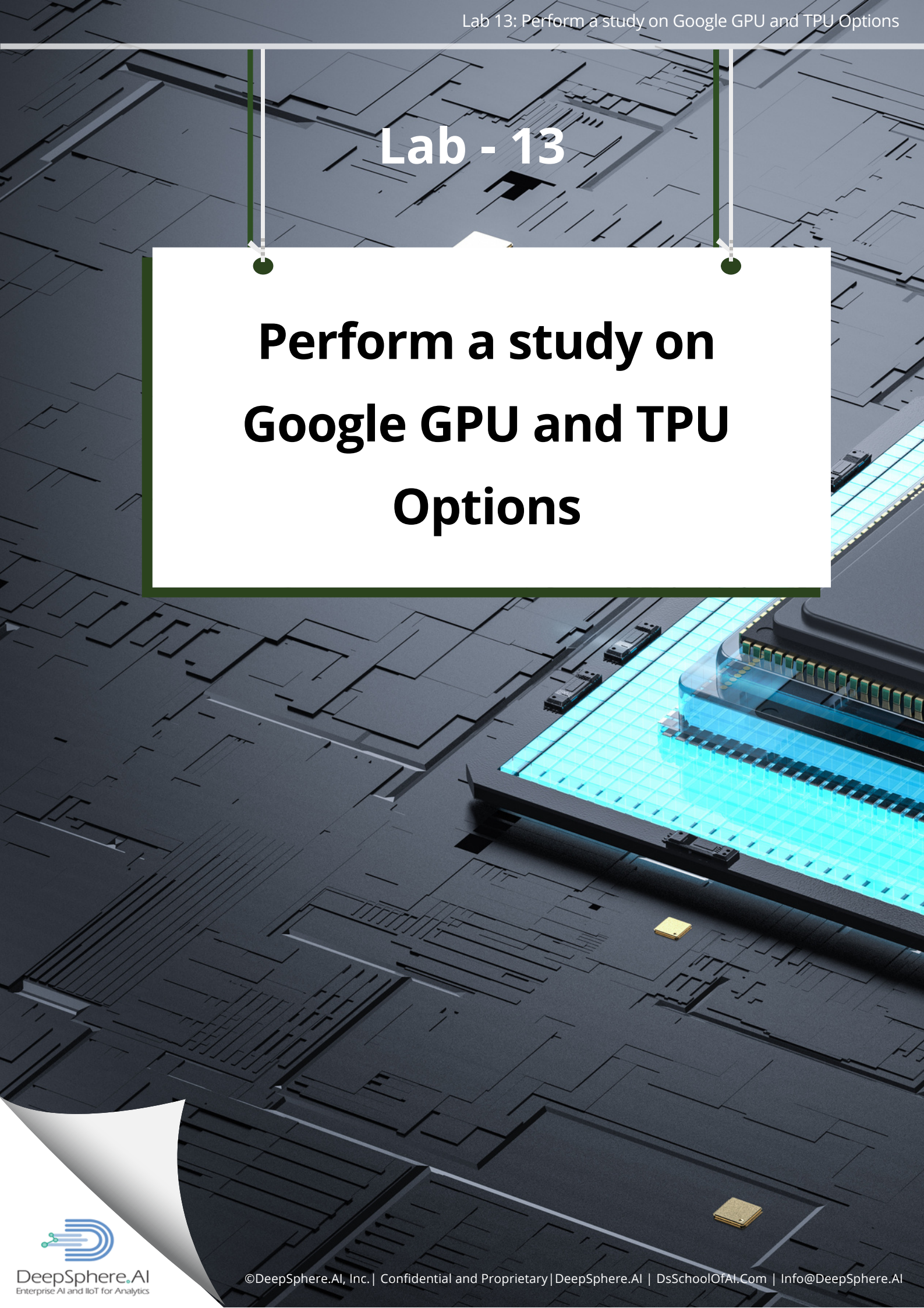
5.**Cloud Spanner:** This service is a globally distributed, horizontally scalable, relational database service. Cloud Spanner provides transactional consistency across multiple regions, making it ideal for applications that require low latency and high availability.

6.Cloud Bigtable: This service is a highly scalable, globally distributed, NoSQL database service. Cloud Bigtable is designed for handling large amounts of data and is optimized for low-latency, high-throughput workloads.

Overall, Google Core Infrastructure Services provide a solid foundation for building and operating applications on Google's cloud platform. They offer a range of options for compute, storage, and databases, making it easy for users to find the right solution for their specific needs. Additionally, these services are built on top of Google's own infrastructure, which is designed for high availability, scalability, and security, giving users confidence in the reliability and security of their applications.

Lab - 13

Perform a study on Google GPU and TPU Options

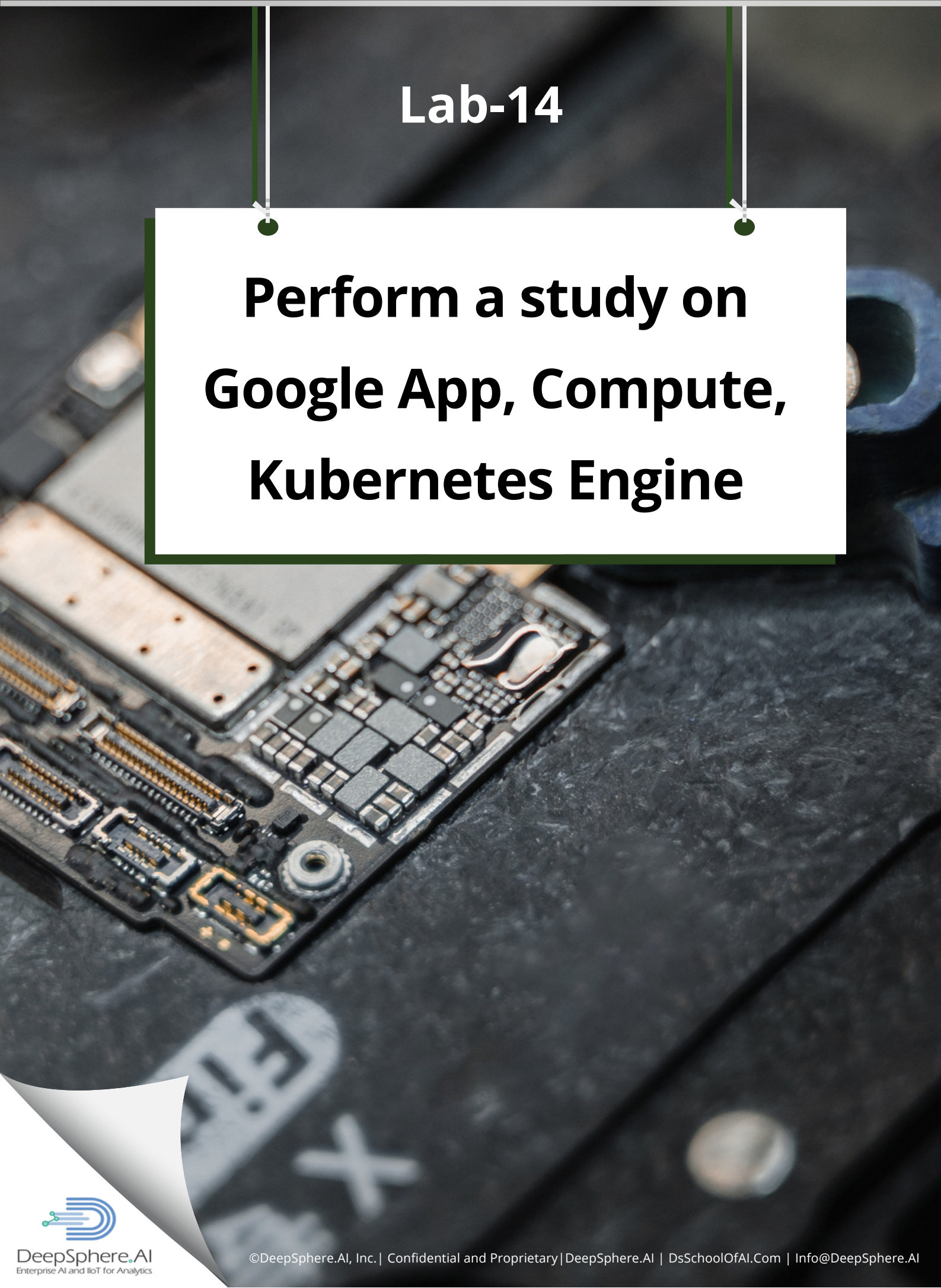


DeepSphere.AI
Enterprise AI and IIoT for Analytics

Perform a study on Google GPU and TPU Options

Google offers a variety of GPU and TPU options for users to choose from when working with machine learning and deep learning applications. These options include:

1. **Google Cloud GPU**: This option allows users to rent GPU resources on the cloud, with support for popular deep learning frameworks such as TensorFlow, PyTorch, and Keras. Google Cloud GPU offers a range of GPU options including NVIDIA Tesla P100, NVIDIA Tesla V100, and NVIDIA Tesla T4.
2. **Google Cloud TPU**: TPU (Tensor Processing Unit) is a custom-built chip designed by Google specifically for machine learning workloads. Google Cloud TPU offers a range of options including TPU v2, TPU v3, and TPU v4. TPUs are designed to perform the matrix multiplications and convolutions required for deep learning at high speed and with low latency.
3. **Google Colab**: This is a free Jupyter notebook environment that allows users to run machine learning and deep learning experiments on GPUs and TPUs. Users can use Colab with a free Google account and have access to a K80 GPU or a TPU for free for a limited time.
4. **Google AI Platform:** This platform allows users to build, deploy, and run machine learning models on the cloud using Google's pre-built machine learning models and services. Users can also use Google AI Platform to train their own models using Google's powerful GPU and TPU resources.



Lab-14

Perform a study on Google App, Compute, Kubernetes Engine

DeepSphere.AI
Enterprise AI and IIoT for Analytics

Perform a study on Google App, Compute, Kubernetes Engine

Google App Engine:

This is a platform for developing and hosting web applications in various programming languages. It includes features such as automatic scaling and load balancing, as well as a variety of runtime environments and support for different languages.

Case study on Google App Engine

Google App Engine is a fully managed platform for developing and hosting web applications in Google-managed data centers. It includes a web application server, a NoSQL datastore, and support for several programming languages and development frameworks. One example of a company using App Engine is the social media platform, Snapchat. They used App Engine to host their backend services and datastore, which allowed them to scale quickly as their user base grew. App Engine's automatic scaling feature allowed Snapchat to handle large spikes in traffic without any manual intervention, and its security features helped keep their data safe. Another example is the online marketplace, Spotify. They used App Engine to host their web services, which handle tasks such as user authentication and playlist management. App Engine's automatic scaling and load balancing capabilities helped Spotify handle the large volume of traffic to their site, and its security features helped protect their user data. App Engine has also been used by many other companies, including NASA, The Guardian, and Khan Academy, to develop and host their web applications. Google App Engine can be used for many different types of applications. It is a suitable option for companies looking to scale their web applications quickly and securely, without the need to manage their own infrastructure.

Case study on Google Compute Engine (GCE):

Google Compute Engine (GCE) is a service provided by Google Cloud that allows users to create and run virtual machines (VMs) on Google's infrastructure. GCE offers a variety of features, including the ability to scale resources up or down based on demand, automatic upgrades and patching, and integration with other Google Cloud services such as storage and networking.

One example of a company that has used GCE is Snapchat. Snapchat used GCE to handle the large amounts of data and traffic generated by its users. By using GCE's auto-scaling feature, Snapchat was able to automatically add or remove servers based on the number of users and their activity. This helped them to reduce costs and improve their service's availability.

Another example is Spotify, they used GCE to power their backend services and process a large amount of data. Spotify used GCE's global load balancing feature to distribute traffic across multiple regions, which helped to improve the service's availability and reduce latency for users.

Additionally, GCE can be used for machine learning and data processing workloads, Spotify used GCE to train machine learning models and process large amount of data.

In summary, GCE offers a variety of features that can help companies to easily and efficiently manage their computing resources, and can be used for a wide range of workloads, including web and mobile applications, data processing, and machine learning. Many companies like Snapchat and Spotify have used GCE to improve their service's availability and reduce costs.

Case study on Kubernetes Engine:

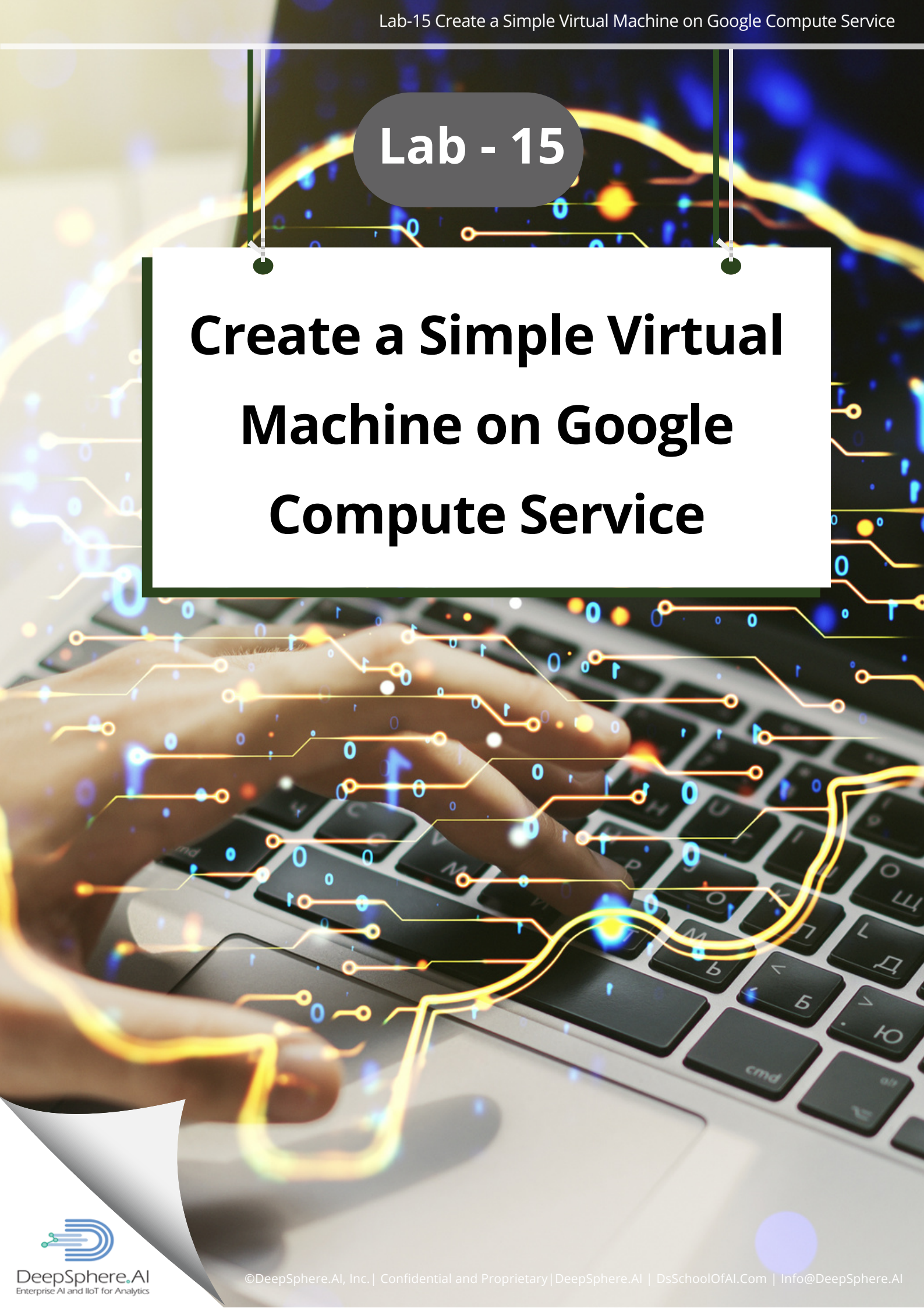
Kubernetes Engine is a powerful platform for managing containerized applications. It is built on top of the Kubernetes open-source container orchestration system, and is designed to make it easy to deploy, scale, and manage containerized applications.

One company that has successfully implemented Kubernetes Engine is a large online retailer. They had been using a traditional monolithic application architecture, but as their business grew, they found that their application was becoming increasingly difficult to manage and scale.

To address these issues, the company decided to move to a microservices architecture, and to use Kubernetes Engine to manage their containerized applications. This allowed them to easily deploy and scale their services, and to automatically handle tasks such as load balancing and automatic failover.

The company also used Kubernetes Engine to manage their containerized databases, which allowed them to easily scale their databases to meet the needs of their growing business.

Overall, the company found that Kubernetes Engine was a powerful and easy-to-use platform that allowed them to easily manage and scale their containerized applications. This helped them to improve their business performance and to better serve their customers.



Lab - 15

Create a Simple Virtual Machine on Google Compute Service

DeepSphere.AI
Enterprise AI and IIoT for Analytics

Create a Simple Virtual Machine on Google Compute Service

Steps:

1. Go to the Google Cloud Console (https://console.cloud.google.com/) and sign in to your Google account.
2. Select the project you want to use for the virtual machine. If you don't have any projects, create a new one.
3. In the sidebar, navigate to the "Compute Engine" section, then select "VM Instances."
4. Click the "Create" button to create a new virtual machine.
5. Fill in the required information, such as the instance name, zone, and machine type.
6. Choose an operating system for the virtual machine. You can either choose from one of the pre-configured images or upload your own.
7. Configure any additional settings, such as firewall rules and access scopes, if desired.
8. Click the "Create" button to create the virtual machine.
9. Once the virtual machine is created, you can connect to it using SSH by clicking the "SSH" button in the VM instances page.
10. You can Start, stop, and manage your instances using the Cloud Console, the gcloud command-line tool, or the API.